Revision and Correction Notes

This special **4.0.1** edition was typeset from a scanned copy of *SPI CCP Version 4.0, Section I* (1994), originally authored and printed using *Microsoft Word for Windows*.

No functional changes have been made to the content. However, several editorial corrections have been applied for clarity and consistency:

- Numerous spelling errors present in the original text have been corrected.
- Compound words not part of the established vernacular of data protocols or plastics have been appropriately hyphenated.
- Capitalization inconsistencies have been resolved while preserving the original's intentional camel-case notation.
- Several punctuation issues—particularly misplaced or missing commas that could cause ambiguity—have been corrected.
- One simple arithmetic error has been corrected.
- One inappropriate superscript on an ordinal number.

Section and table numbering have been verified and matched to the original text, though the depth of the Table of Contents has been reduced for readability.

Finally, the included C source code for the CRC calculation has been cleaned up a bit and made more computer agnostic.

SPI CCP Communication Protocol Documentation
Version 4.0.1 – Section I, Unofficial

SPI Committee on Communication Protocol

1994

Contents

Revision and Correction Notes **1**

1 Introduction . . . 4
1.1 SPI Committee on Communication Protocol Phase I . . . 4
1.2 SPI Committee on Communication Protocol Phase II . . . 4
1.3 SPI CCP Manual Organization . . . 5
2 SPI CCP Protocol Definition . . . 5
2.1 OSI Seven Layer Model . . . 5
2.2 Layer 1, Physical Layer . . . 5
2.2.1 SPI CCP Serial Physical Standard . . . 6
2.2.1.1 Electrical and Procedural Specifications . . . 6
2.2.1.1.1 Signal Transmission Specification, EIA 485 . . . 6
2.2.1.1.2 ANSI X3.15-1976, Bit Ordering . . . 7
2.2.1.1.3 ANSI X3.16-1977, Bit Parity . . . 7
2.2.1.1.4 EIA-404, Baud Rate Tolerances . . . 7
2.2.1.2 Mechanical and Functional Specifications . . . 7
2.2.1.2.1 Network Connector, D Subminiature . . . 8
2.2.1.2.2 Network Cabling . . . 8
2.3 Layer 2, Data Link Layer . . . 8
2.3.1 SPI CCP Serial Data Link Standard . . . 8
2.3.1.1 Establishment and Termination (ANSI X3.28–2.4) . . . 9
2.3.1.1.1 Station Selection Sequence . . . 9
2.3.1.1.2 Polling . . . 10
2.3.1.1.3 Selection . . . 10
2.3.1.1.4 Termination . . . 11
2.3.1.2 Message Transfer Procedure . . . 11
2.3.1.2.1 Transmission Blocks . . . 11
2.3.1.2.2 Replies . . . 14
2.3.1.3 Abort and Interrupt Procedures . . . 14
2.3.1.3.1 Block Abort . . . 14
2.3.1.3.2 Sending Station Abort . . . 15
2.3.1.3.3 Termination Interrupt . . . 15
2.3.1.3.4 Reverse Interrupt . . . 15
2.3.1.3.5 Temporary Interrupt . . . 16
2.3.1.4 Error Detection and Recovery . . . 16
2.3.1.4.1 ERR Error Status Byte . . . 16
2.3.1.4.2 CRC Calculation . . . 17
2.3.1.4.3 Timers . . . 18
2.3.1.4.4 Synchronization . . . 20
2.4 Layer 3, Network Layer . . . 20
2.5 Layer 4, Transport Layer . . . 20
2.6 Layer 5, Session Layer . . . 21
2.7 Layer 6, Presentation Layer . . . 21

2.7.1 SPI CCP Presentation Standard . . . 21
2.7.1.1 SPI CCP Standard Transfer Data Formats . . . 21
2.7.1.1.1 Integral Data Formats . . . 21
2.7.1.1.2 Complex Data Formats . . . 22
2.7.1.2 SPI CCP Device Command Sets . . . 22
2.7.1.2.1 Device Information . . . 22
2.7.1.2.2 Command Information . . . 23
2.7.1.2.3 SPI CCP Required Commands . . . 23
2.8 Layer 7, Application Layer . . . 24
2.8.1 SPI CCP Machine Cell Serial Network Application . . . 24
2.8.1.1 Application Scope . . . 24
2.8.1.2 Network Topology and Underlying Layer Requirements . . . 25
2.8.1.2.1 SPI CCP Serial Physical Standard . . . 25
2.8.1.2.2 SPI CCP Serial Data Link Standard . . . 25
2.8.1.2.3 Network Layer . . . 25
2.8.1.2.4 Transport Layer . . . 25
2.8.1.2.5 Session Layer . . . 25
2.8.1.2.6 SPI CCP Presentation Standard . . . 25
2.8.2 SPI CCP Host Computer Serial Network Application . . . 26
2.8.2.1 Application Scope . . . 26
2.8.2.2 Network Topology and Underlying Layer Requirements . . . 26
2.8.2.2.1 SPI CCP Serial Physical Standard . . . 26
2.8.2.2.2 SPI CCP Serial Data Link Standard . . . 26
2.8.2.2.3 Network Layer . . . 27
2.8.2.2.4 Transport Layer . . . 27
2.8.2.2.5 Session Layer . . . 27
2.8.2.2.6 SPI CCP Presentation Standard . . . 27
3 Appendices . . . 27
3.1 Abbreviations and Acronyms . . . 27
3.2 Glossary . . . 28
3.3 ASCII Character Set . . . 31
3.4 CRC Calculation Program Example . . . 33
3.5 References . . . 34

List of Tables

1 OSI 7-layer stack (illustrative). . . . 6
2 SPI CCP Physical Link Layer Connector (DE-9) . . . 8
3 Station Selection Sequence . . . 9
4 POLL Supervisory Sequence . . . 10
5 SELECT Supervisory Sequence . . . 10
6 SELECT Positive Acknowledgment Sequence . . . 11
7 SELECT Negative Acknowledgment Sequence . . . 11
8 Control Station Message Format . . . 13
9 Tributary Station First Message Format . . . 13
10 Tributary Station Subsequent Message Format . . . 14
11 ERR Error Status Byte Format . . . 17
12 CRC DLE Sequence Chart . . . 18
13 Timer A Functions . . . 19
14 Timer B Functions . . . 19
15 Timer D Functions . . . 20
16 Timer E Functions . . . 20
17 Echo Command Parameters . . . 24
18 Device Revision Command Parameters . . . 24
19 7-bit ASCII Character Set . . . 31

1 Introduction

1.1 SPI Committee on Communication Protocol Phase I

In 1987, a group of SPI Machinery Division members formed a committee to develop a communication protocol standard. In the past, member companies developed various "in-house" protocols to facilitate setup and process data communication between a "primary" plastics processing machine and its various auxiliary devices. Each company had to be compatible with a number of these protocols in order to meet specific vendor requirements. The SPI Committee on Communication Protocol (SPI CCP) was assigned the task to standardize a protocol for use by plastic machinery manufacturers. The committee adopted the following rules to govern the creation of this protocol:

- ***The protocol will be developed using as many existing standards and protocols as possible.*** This rule was designed to help save in development time and add credibility to the final protocol. Also, it will aid in developer and end user understanding as there are existing books and reference documents which describe the various standards and protocols.
- ***The protocol shall use existing and readily available technology and shall not interfere with the normal operations of the processing equipment.*** This rule was designed to help save in development time and add credibility to the final protocol. Also, it will aid in developer and end user understanding as there are existing books and reference documents which describe the various standards and protocols.
- ***The purpose of the protocol is to provide centralized setup and monitoring of various auxiliary equipment by a primary plastics processing machine. The protocol shall not be used for any function necessary for the safe operation of equipment nor perform any direct control of equipment.*** The primary machine would not actually control the auxiliary devices, but would simply allow the machine operator to set an auxiliary machine's operating parameters and then monitor its process data from the primary machine's control station. No direct control functions or function necessary for the safe operation of equipment will be provided by the protocol. At no time will any control or safety function of equipment be dependent on the operation of the communication network.

In 1988 the SPI CCP published the first Communication Protocol Manual. This document described the specific hardware and software requirements to communicate data in a standard format. Along with this document, a test software system was created and distributed to allow a developer to exercise and test an implementation of the protocol without requiring actual primary or auxiliary processing equipment. In the following years auxiliary device command sets were created to standardize on the data transferred between a primary plastic machine and auxiliary devices including mold heaters, chillers, temperature controls feeders, blenders, robots, etc.

1.2 SPI Committee on Communication Protocol Phase II

In 1992 the SPI CCP reconvened to implement Phase 2 of the Communication Protocol. The purpose of this phase was to focus on communication between primary plastics processing machines and a host computer system. At this time the host computer communication environment, like the first phase of the protocol development, the committee operated according to the following rules governing the creation of Phase 2 of the protocol:

- ***The protocol will use as many existing SPI CCP standards, protocols, and procedures as possible.*** By using the same hardware and software interface as used by the original SPI CCP protocol, the Phase 2 host computer interface could be added by simply developing a primary plastics processing machine command set. The existing protocol documentation and test software system would still be applicable. Existing primary plastics processing equipment could be easily retrofitted to provide the host computer interface hardware and software.

- ***The protocol will conform to all original SPI CCP Governing Rules and Design Constraints.*** All of the original SPI CCP governing rules and design constraints must be adhered to including all safety considerations. The main purpose of the protocol will continue to be to provide centralized setup and monitoring of plastics processing equipment. The protocol will not be used to provide direct control of equipment or any function which is necessary for the safe operation of equipment. In all applications of the protocol, the equipment must first ensure that execution of any communicated command or processing of any communicated data is checked for validity and equipment safety standards. The protocol is used only to "request" functions and data transfers, the processing equipment must ensure that the machine is in a safe and valid state before executing the requested function or transferring the requested data. The only change to the original design constraints is the addition of new protocol applications, while maintaining the original machine auxiliary cell network.

In 1994, the SPI CCP introduced Version 4 of the Communication Protocol. This version of the protocol combined the machine cell and host computer network documentation into a single manual. An Injection Molding Machine device command set has been created for this application. Note that the existing SPI CCP protocol versions (V3.0x) have not been changed. The manual organization and presentation have been modified to include support of the phase 2 host computer network but all machine cell auxiliary network functionality has remained the same.

1.3 SPI CCP Manual Organization

The SPI CCP Version 4.00 manual was organized based on the Open Systems Interconnection (OSI) 7 Layer Model. This model provides a basis for developing and discussing the hierarchical layers of communication protocols. The model provides for different standards and protocols for each layer allowing the SPI CCP Protocol to be enhanced in the future to take advantage of newer technologies while maintaining support of existing procedures. The SPI CCP Protocol Definition section describes the standards, procedures, and protocols defined for each of the seven OSI model layers. Each description provides a reference to the existing standards on which the layer is based along with a description of its procedures. It is intended that this document serve as an overview of the various specifications and standards, however, some of the referenced documents may be required for protocol understanding and implementation. Where applicable, multiple (mutually exclusive) standards are presented (which are used in different SPI CCP Protocol applications). The document is concluded with various appendices which provide additional information concerning the protocol.

2 SPI CCP Protocol Definition

2.1 OSI Seven Layer Model

The OSI Seven Layer Model is used to describe and design communication protocols. The model was developed in 1984 by the International Organization for Standardization. The model breaks the overall communication process into 7 hierarchical layers. Standards, procedures, and protocols are then defined for each layer. The compilation of standards for each layer forms an overall computer protocol application. The OSI model was chosen to describe the SPI CCP Protocol as it has become the most common framework used to discuss communication protocols.

2.2 Layer 1, Physical Layer

Layer 1 of the OSI model is the Physical Layer. This layer consists of the electrical and mechanical specifications of the communication link. The layer defines mechanical interface specifications, electrical bit representation specification, bit stream procedures, and transmission activation, maintenance, and deactivation procedures. The collection of these procedures forms the Physical Layer.

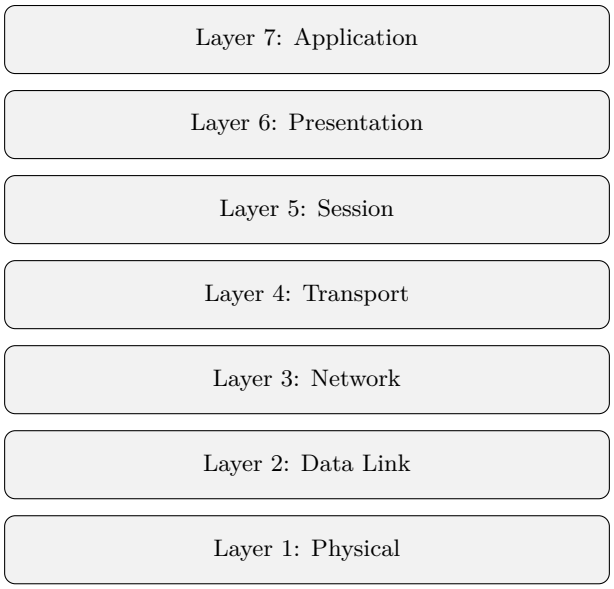

Table 1: OSI 7-layer stack (illustrative).

2.2.1 SPI CCP Serial Physical Standard

The SPI CCP Serial Physical Link Standard is the collection of specifications used to provide a serial based physical layer. This layer was first defined in Version 3.0x of the SPI CCP Protocol. The specification consists of various existing serial communication standards and protocols.

2.2.1.1 Electrical and Procedural Specifications The SPI CCP Serial Physical Link Standard specifies various standards which define the electrical and procedural functions of the transmission of data.

2.2.1.1.1 Signal Transmission Specification, EIA 485 Data communication involves two activities, transmitting data and receiving data. The EIA485 standard is the electrical specification for the various transmitter and receiver circuits that must work in concert to allow for digital data communication to take place. This standard specifies the required voltages, currents, and resistance values measured at the interconnect points and references heavily to EIA422 which will also be discussed.

This standard allows for very high data rates and relatively long interconnect distances. It also specifies that a contention prevention scheme must be employed to keep generated currents from becoming so high as to damage system components. The various electronic devices employed must meet the specification of allowing up to 32 transceiver "loads" on the circuit which will not draw currents that may damage the electrical components. The resistance of the circuit as a whole is also specified to be capable of being as low as 50 ohms. No stub should be greater than 20 m. Sound engineering practice is to keep the stub length at zero, or as close as possible. Systems integrators using more than six 20 m stub lengths in a star topology run the risk of corrupting data. True buss topology with minimum stubs is highly recommended.

Contention occurs when multiple transmitters drive the line simultaneously. While this obviously causes noise on the circuit so as to eliminate the possibility of communication, it also increases the current on the affected circuit. This may occur through a failure in or an initialization of one of the components of the system. Approved circuits must be able to allow for this contention to occur. The software side of this specific protocol dictates that no contention shall occur. The hardware of the communication systems must be designed to allow for some type of current-limiting or thermal shut-down mechanism when contention occurs. It is recommended to utilize current-limiting schemes as they generally allow for a faster recovery period from contention. Sound engineering practices must also be followed on the voltage limiting side as transient over-voltages due to contention interruption may also occur. A passive transmitter must be able to survive 15 µs duration 1% duty cycle 25 volt pulses from a source having an impedance of 100 ohms.

Transmitters must survive under normal conditions the hook-up of all output terminals to any $-7/+12$ volt source under all three output conditions, binary 0, binary 1, or passive. Under the conditions of this test, with the voltage slew rate equal to or less than 1.2 volts/µs the peak current in any lead to the terminals shall not exceed a composite 250 mA for all transmitters.

The concept of the unit load concerns the electrical characteristics of passive transmitters and receivers (are obviously active even when electrically passive) or combination passive transmitter/receivers in the system. The unit load of each communication circuit is obtained by measuring the drawn current under the allowed $-7/+12$ volt range at each receiver pin. It is common practice to utilize chip sets that observe electrical characteristics in the circuit ranging from 0.7 to 1.1 unit loads. This must be specified by each vendor. The total loading on the network of all nodes should not exceed 32 unit loads. Each supplier must supply and preferably certify the data on the communication interface circuit so that the system integrator may not exceed these maximum loading limitations.

The allowable data rates versus distance, as this is a trade-off, are more properly defined in EIA422. This standard states that at the maximum allowable baud rate of our SPI protocol, distances of up to 1200 meters on the circuit without repeaters are supported without distortion with the use of this balanced interface and 24 AWG shielded twisted pair cable with a shunt capacitance of 52.5 pF/M. Greater distances can be achieved through the usage of larger cable sizes while the corollary also applies. A 100 ohm terminating resistor at each end of the buss is recommended. The dispersal of offset grounds, which occurs due to the capacitive coupling between the machine and the hardware/software engine that represents the communication node, under these extreme inter-nodal distances must be allowed for. While optical isolation of all nodes is an excellent form of protection, optionally a resistor in the range of $\frac{1}{2}$ watt and with a resistance of approximately 100 ohms tied from third conductor to logic reference is recommended. This practice also helps eliminate signal distortion caused by electromagnetic interference. Proper operation of the transceiver circuit dictates the presence of a signal return path. This circuit reference is established by the third conductor connecting the common leads of the devices or it may be provided by a connection between earth reference and machine ground. This circuit must contain a $\frac{1}{2}$ watt 100 ohm resistor, if provided.

Additional specified transmitter characteristics are that the non-inverting terminal of the generator shall be negative with respect to the inverting terminal of the transmitter for a binary 1 which is the MARK or OFF state, the non-inverting terminal of the transmitter shall be positive with respect to the inverting terminal of the generator for a binary 0 which is the SPACE or ON state. The receivers should be connected accordingly. The non-inverting terminal should be connected through a 1k–2k ohm resistor to +5 VDC, conversely the inverting terminal should be connected through matching resistance to logic ground within the primary plastics processing machine only.

2.2.1.1.2 ANSI X3.15-1976, Bit Ordering This specification defines the order of bits as they are electrically transmitted. Typically this functionality is provided by a UART device. The SPI CCP Serial Physical Link Layer requires the following settings:

- Word Size: 8 Bits
- Stop Bits: 1 Bit
- Parity: None

2.2.1.1.3 ANSI X3.16-1977, Bit Parity This specification defines the procedure for bit parity calculation and usage. Typically this functionality is provided by a UART device.

2.2.1.1.4 EIA-404, Baud Rate Tolerances This standard specifies acceptable tolerances and other information concerning baud rate timing. Typically this functionality is provided by a UART device. Note that the actual baud rates supported are defined by the protocol application which references this standard as its Physical Link Layer.

2.2.1.2 Mechanical and Functional Specifications The SPI CCP Serial Physical Link Layer calls for various standards and specifications for the mechanical and functional aspects of the protocol.

2.2.1.2.1 Network Connector, D Subminiature A standard connector format has been chosen based upon availability, durability, key-ability, shield-ability, connection/disconnection speed, and expense. The connector is a D-Subminiature. It is multiply sourced, durable, keyed through an asymmetrical pin arrangement, and provides metal shielding for shell ground protection. It can be obtained in fluid resistant configurations and in crimp or solder cup versions. The connector is described in MIL-C-24308. The connector has nine contacts labeled 1–9 and are assigned the following functions:

Contact	Assignment
1	(Third Conductor)
2	(N/C)
3	(N/C)
4	Feed Non-Inverting Circuit
5	Feed Inverting Circuit
6	(N/C)
7	(N/C)
8	Return Non-Inverting Circuit
9	Return Inverting Circuit

Table 2: SPI CCP Physical Link Layer Connector (DE-9)

2.2.1.2.2 Network Cabling As specified by the EIA-485 standard, 24 AWG shielded twisted pair cable with a shunt capacitance of 52.5 pF/m is recommended.

2.3 Layer 2, Data Link Layer

The OSI Data Link Layer defines the methods and procedures used to establish, maintain, and release data link connections. It also specifies the network topology, line discipline, flow control, and error control.

2.3.1 SPI CCP Serial Data Link Standard

The SPI CCP has specified a Serial Data Link Layer protocol to perform the functions required by the OSI Data Link Layer. The protocol references the ANSI X3.28-1976 standard. This standard was the originally specified by V3.0x of the SPI CCP. The ANSI x3.28 standard defines procedures for data communication links using the ANSI X2.4-1968 (ASCII) control characters. The standard provides various methods of establishing a communication link, sending data on the link, and terminating the link. The different procedures are chosen according to the physical layer characteristics and communication system requirements. Specifically, the SPI CCP has chosen subcategory 2.4 for communication establishment and termination procedures and subcategory D1 for message transfer procedures. Together, these subcategories provide the following services and functions:

- **Line Configuration** Subcategory 2.4 provides for a multipoint topology where all stations are interconnected and do not require any connection switching. Subcategory 2.4 also provides for two-way alternate communication which is suitable for half-duplex communication systems where 1 system is designated as the control and is responsible for line discipline. All other stations are designated as tributary stations. This category also provides various methods for terminating communication sessions.
- **Flow Control** Subcategory D1 provides for simple data flow control. Data is broken into blocks and sent one at a time. Each block must be acknowledged before another block may be sent.
- **Error Control** Subcategory D1 provides various techniques for detecting transmission errors. A CRC is used to check all transmitted data characters. Alternating acknowledgments are used to prevent lost or duplicate data blocks. Various timers are used to prevent missed or lost control sequences.

The following sections describe the ANSI x3.28 standard in more detail.

2.3.1.1 Establishment and Termination (ANSI X3.28–2.4) The ANSI x3.28 Subcategory 2.4 defines procedures for communication establishment and termination for two-way alternate, non-switched multipoint, centralized operation communication systems. This subcategory is suitable for communication system configurations where all stations are interconnected, each station can communicate in a half-duplex mode, stations are not required to gain access to the communication link (non-switched), and one system is designated the control station. The ANSI x3.28 Subcategory 2.4 is a protocol for two-way alternate systems with centralized operation. This protocol requires that a single station be designated as the control station while all other stations are designated as tributary stations. The control station is initially assigned the master status and is responsible for starting all communication dialogs. Tributary stations cannot initiate communication on their own nor can they directly communicate with another tributary station.

2.3.1.1.1 Station Selection Sequence To communicate, with a tributary station, the control station, which initial has master status, executes a POLL command to read data from the tributary station or a SELECT command to send data to the tributary station. Both POLL and SELECT requests require a station selection supervisory sequence. This sequence of characters is used to select a specific station to communicate with and indicate the information to be communicated. The SPI CCP has defined the following station selection sequence:

Character	Description
(DEVID)	Device type code
(ADD)	Device address
(CMD1)	SPI CCP Command Code #1
(CMD2)	SPI CCP Command Code #2
(RES)	Reserved by SPI CCP

Table 3: Station Selection Sequence

2.3.1.1.1.1 DEVID The station selection sequence DEVID Device Identification character is used to indicate the type of tributary device being specified by the POLL or SELECT request. Each tributary device type in a particular SPI CCP Serial Data Link Protocol network is assigned a unique device type code. The codes are in the range of `$20-$FF`. The code is used to indicate what POLL and SELECT commands are supported by the device. The device identification codes are assigned by the SPI CCP when a communication application is defined which uses this data link protocol.

2.3.1.1.1.2 ADD The station selection sequence ADD Device Address character is used to select a specific tributary station for communication. The ADD, together with the DEVID, form a unique tributary station identifier. ADD values are in the range of `$20-$FF` and are assigned by the protocol user based on the actual network configuration. Typically, the first instance of a particular device type is assigned the address of `$20`, the second `$21`, and so on.

2.3.1.1.1.3 CMD1/CMD2 The CMD1 Command #1 and CMD2 Command #2 characters are used to indicate a specific command to be executed by a specific tributary device. The command codes are defined by the SPI CCP for each defined tributary device type. Both CMD1 and CMD2 values will be in the range of `$20-$FF`. At the Data Link level, CMD2 is used to indicate the type of command to be executed. If CMD2 is EVEN(bit 0 is reset to 0), the command will be a POLL request. If CMD2 is ODD (bit 0 is set to 1), the command will be a SELECT request.

2.3.1.1.1.4 RES The RES Reserved character is a reserved entry in the station selection sequence. Currently, the SPI CCP specifies that this entry be set to `$20`.

2.3.1.1.2 Polling The control station initiates a POLL request to read data from a tributary station. A POLL request consists of an EOT character followed by a station selection supervisory sequence and ending with an ENQ character as follows:

Character	Description
<EOT>	Resets the communication link
(DEVID)	Specifies the device type
(ADD)	Specifies the device address #1
(CMD1)	Specifies the Command #1
(CMD2)	Specifies the Command #2
(RES)	SPI CCP reserved byte
<ENQ>	Specifies the end of the POLL supervisory sequence

Table 4: POLL Supervisory Sequence

Once the control station sends the POLL supervisory sequence it assumes slave status and expects the polled tributary station to assume master status and begin message transfer. A tributary station detecting its (DEVID) and (ADD) assumes master status and responds in one of two ways:

1. If the tributary station has a message to send it initiates message transfer. The control station assumes slave status.

2. If the station cannot send a message it sends an <EOT>, terminating its master status. Master status reverts back to the control station which may then retry the establishment procedure up to 2 times before notifying its control program of a communication failure. If the control station receives an invalid response or no response to the POLL request it terminates by sending an <EOT>.It may retry the establishment procedure up to 2 times before notifying its control program of a communication link failure.

2.3.1.1.3 Selection The control station initiates a SELECT request to send data to a tributary station. A SELECT request consists of an EOT character followed by a station selection supervisory sequence and ending with an ENQ characters follows:

Character	Description
<EOT>	Resets the communication link
(DEVID)	Specifies the device type
(ADD)	Specifies the device address #1
(CMD1)	Specifies the Command #1 code
(CMD2)	Specifies the Command #2 code (bit 0 will be set to 1)
(RES)	SPI CCP reserved byte, set to `$20`
<ENQ>	Specifies the end of the POLL supervisory sequence

Table 5: SELECT Supervisory Sequence

A tributary station detecting its (DEVID) and (ADD), assumes slave status and sends one of two replies:

1. If the selected station is ready to receive, it sends the following code sequence:

Character	Description
(DEVID)	Repeat of tributary device type
(ADD)	Repeat of tributary device address #1
(CMD1)	Repeat of command #1 code
(CMD2)	Repeat of command #2 code
(RES)	SPI CCP reserved byte, set to `$21`
<DLE>	ACK0 positive acknowledgment
<30>	

Table 6: SELECT Positive Acknowledgment Sequence

Upon detecting this reply, and if the reply is the same as its initial request (DEVID, ADD, CMD1, CMD2 and RES are the same) the control station proceeds with message transfer.

2. If the selected station is not ready to continue because the command requested is not a valid command the selected station responds with the following code sequence:

Character	Description
(ERR)	SPI CCP Error Code which indicates the nature of the request refusal
<NAK>	Negative acknowledgment

Table 7: SELECT Negative Acknowledgment Sequence

If the control station does not receive a response or the control station receives an invalid response (DEVID, ADD, CMD1, CMD2 or RES do not match or too many characters are received), it may retransmit the selection supervisory sequence up to 2 times before notifying its control program of a communication link failure and then executing a sending station abort.

If the control station receives a negative acknowledgment response ((ERR)<NAK>), indicating that a character may have been garbled during transmission, it may retransmit the selection supervisory sequence up to 2 times before notifying its control program of a communication link failure and then executing a terminate function.

2.3.1.1.4 Termination Once a communication sequence has been established and the message transfer procedure has completed, the master station transmits an <EOT> to indicate that it has no more messages to send. The <EOT> negates the master/slave status of both stations and returns master status to the control station.

2.3.1.2 Message Transfer Procedure The ANSI x3.28 Subcategory D1 is used as the message transfer procedure for the SPI CCP Data Link Layer protocol. This standard is compatible with the Subcategory 2.4 establishment and termination procedure and provides message independent blocking, cyclic checking, alternating acknowledgments, and transparent heading and text. Message data can be any ASCII value (`$00-$FF`). Data can be sent in variable length blocks (subject to a maximum block length which is set by the communication application which uses this protocol as its data link layer) regardless of the content of the data. Each block is acknowledged with an alternating acknowledgment to allow detection of missed or duplicate blocks and the block data is validated using a Cyclic Redundancy Check (CRC) calculation.

2.3.1.2.1 Transmission Blocks The transmission of blocks is initiated by the master station after an appropriate establishment procedure. If the block is the first block within a message it is started with the code sequence <DLE><SOH>. If the block is not the first block of a message it is started with the code

sequence <DLE><STX>. A block that ends at an intermediate point within the message is ended with the code sequence <DLE><ETB> immediately followed by a cyclic redundancy check sequence (CRC)(CRC). A block which ends at the end of a message is ended with the code sequence <DLE><ETX> immediately followed by a cyclic redundancy check sequence (CRC)(CRC). After the (CRC)(CRC) sequence the master station waits for a reply. Since any of the 256 combinations of 8 bits may occur in the data, the stations must be able to distinguish between control characters and data. The control characters that can be recognized must always be a character sequence started with a <DLE>. When a <DLE> occurs in the source data stream, the master station inserts an additional <DLE> so that the data transmitted will contain two successive <DLE> characters.

The slave station must scan the incoming data for <DLE> characters. On the first appearance of a <DLE>, the character immediately following determines which of the following actions to be taken:

1. If the character immediately following is a <SOH>, the slave station enters transparent mode. The data following is considered the message heading. The message heading is terminated by the code sequence <DLE><STX>.

2. If the character immediately following is a <STX>, the slave station enters transparent mode if not already in transparent mode. The data following is considered as message text. The message text is terminated by the code sequences <DLE><ENQ>, <DLE><ETB>, <DLE><ETX>, the slave station then exits transparent mode.

3. If the character immediately following is also a <DLE>, indicating a data character, the slave station discards one <DLE>. The character immediately succeeding is not considered part of any control character sequence. The station continues to receive data.

4. If the character immediately following is either <ENQ>, <ETB>, or <ETX>, the slave station leaves transparent mode and performs the action designated by the control character (completion of a time-out (timer A, B or D) also causes the slave station to leave transparent mode).

5. If the character immediately following is a <SYN>,the slave treats this as an idle character sequence and discards both characters.

6. If the character immediately following is other than <SOH>, <STX>, <DLE>, <ENQ>, <ETB>, <ETX> or <SYN> then it is to be considered invalid and the entire block is to be considered invalid and must be responded to as such.

2.3.1.2.1.1 Control Station Message Format The control station sends messages when it is executing a SELECT sequence. The following is the control station message format:

Character	Description
<DLE>	Start of Text sequence
<STX>	
(DATA)	Start of data bytes. Can be any value (`$00-$FF`). <DLE> data values are replaced with a transparent DLE sequence (<DLE><DLE>). Data can be any length up to the maximum data block size (TDLE's count as a single data character).
<DLE>	End of Block sequence
<ENQ> or	Block Abort ending sequence
<ETB> or	End of intermediate block sequence
<ETX>	End of final block sequence
CRC1	MS Byte of CRC calculation
CRC2	LS Byte of CRC calculation

Character	Description

Table 8: Control Station Message Format

2.3.1.2.1.2 Tributary Station Message Format A tributary station sends messages when it is executing a POLL request. If the tributary is sending the first block of a message, the following format is used:

Character	Description
<DLE>	Start of Header sequence
<SOH>	
(DEVID)	Repeat of POLL request DEVID
(ADD)	Repeat of POLL request ADD
(CMD1)	Repeat of POLL request CMD1
(CMD2)	Repeat of POLL request CMD2
(RES)	Repeat of POLL request RES
(ERR)	SPI CCP Error Status Byte, set to `$20` if no errors, otherwise error bits are set accordingly
<DLE>	Start of Text sequence
<STX>	
(DATA)	Start of data bytes if data is required. Can be any value (`$00-$FF`). <DLE> data values are replaced with a transparent DLE sequence (<DLE><DLE>). Data can be any length up to the maximum data block size (TDLE's count as a single data character). If no data is required, then no (DATA) bytes are sent between the Start of Text and End of Block sequences.
<DLE>	End of Block sequence
<ENQ> or	Block Abort ending sequence
<ETB> or	End of intermediate block sequence
<ETX>	End of final block sequence
CRC1	MS Byte of CRC calculation
CRC2	LS Byte of CRC calculation

Table 9: Tributary Station First Message Format

If the tributary is sending a subsequent block of a message, the following format is used:

Character	Description
<DLE>	Start of Header sequence
<STX>	
(DATA)	Start of data bytes if data is required. Can be any value (`$00-$FF`). <DLE> data values are replaced with a transparent DLE sequence (<DLE><DLE>). Data can be any length up to the maximum data block size (TDLE's count as a single data character). If no data is required, then no (DATA) bytes are sent between the Start of Text and End of Block sequences.
<DLE>	End of Block sequence
<ENQ> or	Block Abort ending sequence
<ETB> or	End of intermediate block sequence

Character	Description
<ETX>	End of final block sequence
CRC1	MS Byte of CRC calculation
CRC2	LS Byte of CRC calculation

Table 10: Tributary Station Subsequent Message Format

2.3.1.2.2 Replies The slave station, upon detecting the code sequence <DLE><ENQ>, <DLE><ETB>, or <DLE><ETX> followed by the (CRC)(CRC) sequence, sends one of two replies:

1. If the transmission block was accepted and the slave station is ready to receive another block, it sends the appropriate <ACKn>. <DLE><31> is used as the reply to the first block received. <DLE><30> is used to reply to the second block received. Subsequent positive acknowledgments alternate between <ACK1> and <ACK0>. Upon detecting the correct <ACKn>, the master station may either transmit the next block,or initiate termination if the last block was ended in <DLE><ETX>.

2. If the transmission block was not accepted and the slave station is ready to receive another block, it sends the code sequence (ERR)<NAK>, where (ERR) indicates why the block was not accepted. Upon detecting the <NAK> the master station initiates retransmission of the last block up to 2 times before notifying its control program of a communication link failure and initiating a termination function.

The use of a (ERR)<NAK> sequence does not alter the sequence of alternating acknowledgments. The same affirmative reply (<ACK1> or <ACK0>) is used for a successful retransmission as would have been used if the previous transmission of the unaccepted block had been successful.

If the alternating reply indicates that the slave station missed the outstanding block (receipt of <ACK1> instead of <ACK0>,or vice versa), the master station initiates retransmission of that block as if the slave station had returned a <NAK>.

If the master station receives an invalid reply or no reply, it may send a reply request sequence consisting of an <ENQ>. The master station may retry up to 2 times before notifying its control program of a communication link failure and initiating a sending station abort.

Upon receipt of a reply request sequence the slave station must retransmit its last reply. This may be done up to 2 times before notifying its control program of a communication link failure and initiating a terminate function.

2.3.1.3 Abort and Interrupt Procedures The ANSI x3.28 standard provides various methods of aborting and interrupting communication sessions. The following sections describe these procedures.

2.3.1.3.1 Block Abort The Block Abort procedure is used when the sending station is in the process of sending a block and is required to terminate the transfer before the normal end of the block such that the receiving station will discard the block.

2.3.1.3.1.1 Application The Block abort procedure may be used by the sending station when, in the process of sending data, there occurs an indication that the data being sent may not be valid. Such an indication would result from an error condition being detected in the data being sent or of insufficient data being available to send or of excessive amounts of data being available to send. This condition could be the results of a software error or operator error. Block aborts may be used in the message transfer state to cause a temporary text delay after the receipt of the previous acknowledgment if the sending station is not capable of transmitting the text of the next transmission block before the predetermined time out period.

2.3.1.3.1.2 Procedure The Block Abort Procedure is accomplished by the sending station's ending the block at any time by transmitting the code sequence <DLE><ENQ>. The sending station then halts transmission and waits for a reply from the receiving station. The receiving station detects that the block was ended with the code sequence <DLE><ENQ> rather than with the normal ending code sequence

<DLE><ETX> or <DLE><ETB>, whereby the receiving station discards that portion of the block that had been received, and sends a <NAK> with the required (ERR) prefix and remains in the receive state. Following the receipt of the (ERR)<NAK> response the sending station will re-initiate the transmission of the faulty block or initiate the appropriate termination or recovery procedure. Note that if the faulty block cannot be retransmitted then the sending station must initiate the recovery procedure. In the case that a (ERR)<NAK> sequence is not received by the sending station, the sending station will time out and then will normally send an <ENQ> code to request a retransmission of the last response . However, due to a limited number of allowable retries (2) the sending station will eventually execute a recovery procedure.

2.3.1.3.2 Sending Station Abort The Sending Station Abort procedure is used by the sending station to prematurely terminate transmission of a multi block message. After an intermediate block has been sent and a valid acknowledgment has been received (either ACKn or NAK), the sending station can issue a Sending Station Abort to terminate further message transfer.

2.3.1.3.2.1 Application Sending Station Abort procedures may be used by a sending station when in the process of sending a block of text it determines that it should prematurely terminate transmission to the particular receiving station. Such a determination would be caused by the need of a higher priority message to be transmitted to the same or another station, or the inability of the sending station to continue transmission. Sending station abort procedures may be used following block abort procedures to accomplish a transmission abort condition; that is the sending station prematurely terminates the transmission within a transmission block.

2.3.1.3.2.2 Procedure Sending station abort procedures are accomplished by the sending station's completing the transmission of a block by sending the code sequence <DLE><ENQ> or <DLE><ETB>. Then upon receipt of the proper acknowledgment code sequence <ACKn> or (ERR)<NAK>, or no acknowledgment, the sending station transmits an <EOT> to terminate the transmission to the particular receiving station. The receiving station detects this sending station abort procedure by recognizing receipt of the <EOT> following the code sequence <DLE><ENQ> or <DLE><ETB> instead of the normal code sequence <DLE><ETX>. Receipt or transmission of <EOT> resets all stations on the link to the control phase in nontransparent mode. At this time the supervisory station (control station) will regain control of the communication link. Any data received by the receiving station during this message transfer prior to the sending station abort procedure will be discarded.

2.3.1.3.3 Termination Interrupt The Termination Interrupt procedure is used by the receiving station after receipt of a message or transmission block to terminate further transmission by the sending station.

2.3.1.3.3.1 Application The termination interrupt may be used by the receiving station to cause the transmission to be interrupted because it is not in a condition to receive. Such a condition could exist because the receiving station is off line or has been requested to disallow any communications.

2.3.1.3.3.2 Procedure The termination interrupt procedures are accomplished by the receiving station transmitting an <EOT> in lieu of one of its normal acknowledgments such as <ACKn> or (ERR)<NAK> or any other valid response. This response indicates a negative acknowledgment of the last transmission and the conclusion of the transmission. This <EOT> resets all stations on the link to the control phase in nontransparent mode. At this time the supervisory station (control station) will regain control of the communication link.

2.3.1.3.4 Reverse Interrupt The Reverse Interrupt procedure is used by a receiving station to request the sending station to terminate the transmission in progress prematurely in order to facilitate a reversal in the direction of data transfer.

2.3.1.3.4.1 Application Reverse interrupt procedures may be used by the control station to interrupt its receiving of a "batch" message stream so that it may transmit a priority message 'to the original sending station or to another station. This function may only be used by the supervisory station (control station).

2.3.1.3.4.2 Procedure Reverse interrupt procedures may be used by the control station only after reception of a block with a valid cyclic redundancy check (CRC). Reverse interrupt procedures are accomplished by the control station transmitting the code sequence <DLE><3C> in lieu of the normal affirmative acknowledgment <ACKn>. This reply is interpreted as an affirmative acknowledgment by the receiving station to the last transmitted block, and it signals a request by the control station that the sending station terminate the transmission sequence in progress as soon as the sending station is in such a state that it can receive a message without destroying or loosing information that had previously been stored in buffers. The <DLE><3C> code sequence cannot be retransmitted by the control station to successive transmission blocks without transmitting intervening affirmative acknowledgments (<ACKn>). However, successive <DLE><3C> code sequences can be transmitted for a given block in response to <ENQ>'s being received for retry purposes. Upon receipt of the code sequence <DLE><3C> the transmitting station should terminate the transmission by transmitting an <EOT> after it has completed transmitting all data that would prevent it from receiving a message. All transmitting stations must limit the number of blocks that they must transmit to a maximum of 1. The receipt of the code sequence <DLE><3C> as a response to a sending station <ENQ> should be treated as a repeated response if the last valid response received was <ACKn>. The sending station should continue by transmitting the next block or <EOT>. If the last valid response was the code sequence <DLE><3C> then the transmitting station should assume that the last block was received in error and retransmit the last block.

2.3.1.3.5 Temporary Interrupt The ANSI x3.28 Temporary Interrupt (Wait Acknowledgment) procedure is specified as optional and is not used by the SPI CCP Data Link protocol.

2.3.1.4 Error Detection and Recovery The ANSI x3.28 standard provides various methods of error detection and recovery. Header data is duplicated in reply messages to ensure the proper station has responded. A CRC calculation is used to protect against lost or garbled data bytes. And various timers are used to protect against lost control characters and no responses. The following sections describe the ANSI x3.28 error detection and recovery procedures in greater detail.

2.3.1.4.1 ERR Error Status Byte The SPI CCP ERR Error Status byte is used to report error information for refused commands or data blocks. The CPI CCP ERR value is returned in tributary station POLL replies, control station POLL data block acknowledgments, and slave station SELECT replies and acknowledgments. The ERR value is defined as a BIT register where each bit is used to indicate a specific status as follows:

Bit	Description
7 (MS bit)	**Invalid Data** When set to 1, the receiving station has detected invalid data. Data has been refused and discarded.
6	**Command Not Ready** When set to 1. the requested command cannot be executed at this time. They command may be attempted again at a later time.
5	**Set to 1** This bit is always set to 1 to ensure that the ERR value is in the range from `$20-$FF`.
4	**Reserved by SPI** This bit is reserved by SPI CCP and must be reset to 0.
3	**Command Not Supported** When set to 1, the received command is not supported by the specified device/address. Command may be retried to determine if request header data was garbled.

Bit	Description
2	**Command Not Executed** When set to 1, the requested command was not executed. Typically this bit is set along with one of the error indication bits.
1	**Invalid Preamble (Supervisory Sequence)** When set to 1, the received command supervisory sequence data was invalid. The request should be attempted again.
0 (LS bit)	**Communication Error** When set to 1, a communication error, such as a CRC error or physical link error, has been detected by the receiving station. The request or data block should be resent.

Table 11: ERR Error Status Byte Format

2.3.1.4.1.1 Invalid Data Bit When this bit is set to 1 it means that the data received was received "correctly", but that it was not appropriate for the command used. Reasons for this for example, would be data out of range, too little data, too much data or an undetected CRC error which produced erroneous data. This bit is set during the negative reply to any block where invalid data is detected.

2.3.1.4.1.2 Command Not Ready Bit When this bit is set to 1 it means that the requested command cannot be executed at this time. The control station should attempt the request again at a later time. No specific time period is specified at the Data Link level.

2.3.1.4.1.3 Command Not Executed Bit When this bit is set to 1 it means that the command requested was not executed or no action was taken. Reasons for this, for example, would be that the controller has been told not to perform the selected command, that the controller has determined that executing the command would result in an unsafe condition within the selected equipment, that the data was invalid, that the preamble was invalid or that there was a communication link error. This bit is set during the negative reply to a block which contains data that would prevent the command from being executed.

2.3.1.4.1.4 Invalid Preamble Bit When this bit is set to 1 it means that the preamble was in error, such as when an invalid RES byte or invalid preamble length is received. This condition could result from a communication link error. This bit is set during the negative reply to any block which has an invalid preamble.

2.3.1.4.1.5 Communication Error Bit When this bit is set to 1 it means that the communication link experienced an error in its communication. This could result from a framing error, a CRC error being detected or a protocol violation. This bit is set during the negative reply to any block in which a communication error was detected.

2.3.1.4.1.6 Command Not Supported Bit When this bit is set to 1 it means that the requested command is not supported by the controller. Each tributary device will return this bit set in the ERR byte (possibly with other bits set such as Command Not Executed) when an unsupported command is received. If the ERR byte is returned with this bit set, the control station should retry the POLL or SELECTION to verify that the tributary device properly received the preamble data. This bit is set during the negative reply to any block which has an unsupported command.

2.3.1.4.2 CRC Calculation The ANSI x3.28 Message Transfer Subcategory D1 requires the use of cyclic redundancy check characters to detect against garbled, lost, or duplicate data bytes. The CRC specified by the SPI CCP Data Link Layer protocol is based on a polynomial of the 16th degree as follows:

$CRC_{16} = x^{16} + x^{15} + x^2 + 1$

The CRC is initiated by the occurrence in the data stream of either <DLE><SOH> or <DLE><STX>. The CRC is signaled to stop through the occurrence in the data stream of <DLE><ETB>, <DLE><ETX>, or <DLE><ENQ>. The following table governs the CRC calculation in deciding which characters to process:

Table 12: CRC DLE Sequence Chart

Character Sequence	Included in CRC	Not Included in CRC
<DLE><SYN>	—	<DLE><SYN>
<DLE><SOH>	—	<DLE><SOH>
<DLE><STX>	—	<DLE><STX>
<DLE><STX> [1]	<STX>	<DLE>
<DLE><ETB> [2]	<ETB>	<DLE>
<DLE><ETX>	<ETX>	<DLE>
<DLE><DLE>	<DLE> (one)	<DLE> (one)

[1] If not preceded in same block by transparent header data.
[2] If preceded in same block by transparent header data.

The bits of the transmission block or message that are included in the CRC calculation, which may include both information as well as control characters, correspond to the coefficients of a message polynomial having terms from $x(n-1)$ down to x^{16}, where x is the total number of bits in the block or message. The resulting polynomial is divided modulo 2 by the generating polynomial. The check bits correspond to the coefficients from the terms from x^{15} to x^0 in the remainder polynomial found when the resulting division has completed. The complete block, which consists of the data followed by the CRC corresponds to the coefficients of a polynomial that is integrally divisible in modulo 2 fashion by the generating polynomial. At the transmitting end, the data bits are subjected to the encoding process which is equivalent to a division by the generating polynomial. The results are transmitted immediately following the data bits as delimited by <DLE><ETB> or <DLE><ETX>, commencing, as previously mentioned, by the highest order bit of the CRC to the lowest, sequentially.

At the receiving end, the incoming block is decoded by a process equivalent to a division by the generating polynomial. If no errors occur, the results are a zero remainder. If the results of this process are other than zero, an error(s) is indicated. This process may be carried out by the use of a 16-stage cyclic shift register with proper feed-back gates which must be all set to zero before processing each block. At the receiver, a zero after decoding would indicate a reception of data without errors. The CRC has a full 16-bit range, so the receiver must not try to interpret the CRC data as control codes.

2.3.1.4.3 Timers The ANSI x3.28 standard specifies the use of various timers for the protection of lost or unrecognized control characters and control character sequences. The following sections describe each of the protocol timers.

2.3.1.4.3.1 Timer A, Response Timer Timer A is used by the sending station to protect against an invalid response or no response at all. Its value is 1000 ms. The following table specifies the use of Timer A.

Timer A Function	Description
Start	Started after transmission of ending sequence where a response is expected such as <DLE><ETB>, <DLE><ETX> or <ENQ>.
Stop	Stopped on receipt of valid reply such as <ACKn>, (ERR)<NAK> or <EOT>.

Timer A Function	Description
Time-Out	Upon time-out: 1. Transmit the reply request <ENQ> character. 2. After attempting reply request two times, execute the sending station abort and return to transparent mode. 3. Notify the control program of a communication failure.

Table 13: Timer A Functions

2.3.1.4.3.2 Timer B, Receive Timer Timer B is used by a receiving station to protect against nonrecognition of end-of-block or end-of-text sequences by a receiving station. Its value is 100 ms. The following table specifies the use of Timer B.

Timer B Function	Description
Start	Started upon receipt of start-of- block sequence such as <DLE><SOH> or <DLE><STX>.
Restart	Restarted upon receipt of any character.
Stop	Stopped on receipt of valid terminating sequence such as <DLE><ENQ>, <DLE><ETB>, or <DLE><ETX>.
Time-Out	Upon time-out: 1. Prepare to receive another transmission. 2. Notify the control program of a communication link failure and discard the data block. 3. Return to non-transparent mode.

Table 14: Timer B Functions

2.3.1.4.3.3 Timer C, Gross Timer This timer is not used by the ANSI x3.28 Subcategories 2.4 and D1.

2.3.1.4.3.4 Timer D, No Activity Timer Timer D serves as a no activity time-out for all stations. Its value is 1200 ms. The following table specifies the use of Timer D.

Timer D Function	Description
Start	Started upon receipt or transmission of any character.
Restart	Restarted upon receipt or transmission of any character.
Stop	Stopped upon transmission or receipt of <EOT>.

Timer D Function	Description
Time-Out	Upon time-out: 1. If receiving station, notify the control program. 2. If supervisor station, return to control mode and notify control program. 3. Return to nontransparent mode.

Table 15: Timer D Functions

2.3.1.4.3.5 Timer E, Delay Before Transmission Timer E is specified by the SPI CCP to enforce a line - turnaround delay at the Data Link layer. Timer E is used by a station to delay transmitting when it is assigned transmit status to allow the previous transmitter time to prepare to receive data. The following table specifies the use of Timer E.

Timer E Function	Description
Start	Started upon assignment of transmit status.
Stop	Timer E is a delay timer and is never stopped.
Time-Out	Time-Out Upon time-out, the delaying station may begin transmitting.

Table 16: Timer E Functions

2.3.1.4.4 Synchronization The synchronization procedure is used as a time-fill in order to prevent timer A and timer D from timing out because the transmitting station is currently unable to continue transmission due to a higher priority task which is running in its control program. This delaying tactic should be performed only for short periods of time. During non transparent mode <SYN> control codes may be arbitrarily added at any point in the transmission. During transparent mode <DLE><SYN> control code sequence may be arbitrarily added at any point in the transmission except:

1. In a <DLE> sequence.
2. Between <DLE><ETB> or <DLE><ETX> and the following cyclic redundancy check (CRC)(CRC) sequence.

The control characters <SYN> or <DLE><SYN> must be discarded by the receiving station.

2.4 Layer 3, Network Layer

The OSI Model Network Layer is responsible for establishing, maintaining and — terminating network connections. The layer provides the mechanisms to communicate with connected networks and subnetworks.

Currently, the SPI CCP has not defined any OSI Model Network Layer protocols.

2.5 Layer 4, Transport Layer

The OSI Model Transport Layer is responsible for transferring data between the upper and lower layers of the OSI model. The layer prevents the upper layers from requiring details on how data is communicated by the lower layers. Currently, the SPI CCP has not defined any OSI Model Transport Layer protocols.

2.6 Layer 5, Session Layer

The OSI Model Session Layer provides a variety of services and functions which are used by the application layer to access the communication network. Such services include the establishment of session connections, data exchange, data synchronization, semaphore processing, and other functions. Currently, the SPI CCP has not defined any OSI Model Transport Layer protocols.

2.7 Layer 6, Presentation Layer

The OSI Model Presentation Layer is concerned with the transfer syntax used between communicating systems. The layer requires that communicating devices determine and use a compatible transfer syntax to exchange application data.

2.7.1 SPI CCP Presentation Standard

The SPI CCP has defined various procedures and standards used to satisfy the OSI Model Presentation Layer requirements. This layer defines the abstract syntax used to represent data to be communicated as well as the format of the data communicated. To provide these data exchange services, the SPI CCP has defined standard transfer data formats and device command sets.

2.7.1.1 SPI CCP Standard Transfer Data Formats The SPI CCP has defined several standard transfer data formats. These formats are used to transfer data between systems which may represent the data in different application formats. Defined integral formats include Numeric, Status, and Open data types. Various complex formats have also been defined which include combinations of integral formats as well as application specific data types.

2.7.1.1.1 Integral Data Formats All data communicated by an application which uses the SPI CCP Presentation Layer protocol will be defined in one of three integral data formats. The Numeric data type is used to transfer all numeric information. The Status data type is used to transfer status and set information. The Open data type is used to transfer all other application data.

2.7.1.1.1.1 Numeric Data The Numeric data type is used to represent all continuous value data. Examples of numeric data include process setpoint values, process actual values, process data limit values, etc. Numeric data is communicated in IEEE754-1985 Single Precision Floating Point notation. Numeric data is converted from the application's internal format to a floating point value for communication. Numeric data values are presented to lower OSI Model layers as a series of 4 bytes with the first byte to be sent consisting of the floating point value's most significant byte.

2.7.1.1.1.2 Status Data Status data is used to represent all information which is discrete (discontinuous) in value. Examples of status data include operating modes, alarm indications, device status information, etc. Typically, status data information is represented in two ways:

1. Status Bit information is used to represent binary data values such as ON/OFF, TRUE/FALSE, OK/ERROR, etc.
2. Status Mode information is used to allow a group of bits to represent entries of a set of information. The bits are grouped as a binary number from 0 to $2^n - 1$ where n specifies the number of bits in the group.

Status data is presented to lower OSI model layers as a series of 2 bytes. The status word most significant byte is communicated first.

2.7.1.1.1.3 Open Data The Open data type is used to represent all information which cannot be expressed as numeric or status data. Such data includes application file data, text strings, etc. The Open data type is also used to communicate complex data types such as Blanket Data Lists.

Open data type data is presented to lower OSI Model layers as a series of bytes. Each defined Open data value must specify the length to be communicated. Open data values can be defined to have both static and dynamic lengths.

2.7.1.1.2 Complex Data Formats The SPI CCP Presentation layer defines various standard complex data formats which can be used in the transfer of data. Although the data will be cast as an Open data value to lower OSI Model layers, the array of Open data bytes are given a specific format.

2.7.1.1.2.1 Blanket List Data The Blanket List complex data type is used to communicate a fixed list of defined integral data values. The purpose of the Blanket List data type is to allow the single integral data values to be communicated in a single request. A Blanket List contains a series of data value entries. Each entry consists of the Command #1 and Command #2 byte values assigned to the data value followed by the actual data in its integral format.

2.7.1.2 SPI CCP Device Command Sets The SPI CCP uses Device Command Sets to define the transfer syntax supported by specific devices. The defined device command sets allow a control station to communicate with various tributary devices knowing only the types of devices connected and without knowledge of the device manufactures. A command set is created for each device type assigned a Device Identification (DEVID) code by the SPI CCP. A command set consists of a collection of POLL (read) and SELECT (write) commands. The commands are used to read and write specific data supported by the device. All commands marked as required will always be supported by devices of the specific DEVID type. Commands marked as optional may or may not be supported by the device. The SPI CCP forms subcommittees to develop command sets for specific device types. Command sets can also be developed for devices not considered by the SPI CCP and vendor specific commands can be added to SPI CCP command sets. The following sections provide the guidelines and practices required to develop or add to a device command set.

2.7.1.2.1 Device Information Each device to be connected to a control station via a communication network is required to define various information used to identify the device and define the devices communication syntax.

2.7.1.2.1.1 Device Name Each device must be assigned a unique device name. A device name consists of an 8 character ASCII string. If less than 8 characters, the device name should be append to 8 characters with NULL (`$00`) characters.

2.7.1.2.1.2 Device Identification Code Each device type is required to have a unique Device Identification Code (DEVID). A DEVID is a byte value in the range of `$20-$FF`. The SPI CCP has reserved DEVID's in the range of `$20-$7F` for SPI CCP approved devices. All other DEVID values (`$80-$FF`) can be used for non SPI CCP approved devices. Note that non SPI CCP approved devices must follow all guidelines for the creation of command sets as specified by the SPI CCP. Also, non-approved device command sets should be presented to the SPI CCP for assignment of an approved DEVID.

The DEVID value is used to identify a specific device type. The DEVID, along with the Device Address (ADD), is used by the lower network layers to select a single specific device for communication. A device which responds to requests of a specific DEVID must support all required commands of the DEVID's command set.

2.7.1.2.1.3 Device Revision Level Each device must define a Device Revision Level. This consists of 4 ASCII characters in the format XX.YY where XX denotes the major revision level (set by the SPI CCP) and YY denotes the minor revision level (set by the particular device implementation).

2.7.1.2.1.4 Device Description Each device must contain a description of its use and function.

2.7.1.2.2 Command Information A list of commands is created for each unique device type. The command list defines the syntax of the data which can be communicated by the device. Each command in list is assigned a name, codes, data type information, support requirement, and description.

2.7.1.2.2.1 Command Name Each command of a particular command set is assigned a unique command name. The command name is an 8 character ASCII string. If less then 8 characters, the string is appended with NULL characters.

2.7.1.2.2.2 Command Codes Each command is assigned a unique Command Code. A Command Code consists of a CMD1 byte and a CMD2 byte. The CMD1 byte is used to group similar commands while the CMD2 byte is used to specify a particular command within the group. For each CMD1 value, two CMD2 values must be reserved. The CMD2 value with bit 0 reset to 0 will be used for POLLing while the CMD2 value with bit 0 set to 1 will be used for SELECTion. When a command is defined, it must specify whether POLLing, SELECTion, or both are supported. Devices which present data based on multiple zones within a single device can use CMD1 as a Zone Selection value. A single command can be defined which specifies the information for a CMD2 POLL/SELECT pair. CMD1 can then be varied across a defined range to access the data for a particular zone. The valid CMD1 zone range can be defined as a constant range or determined dynamically via another command.

2.7.1.2.2.3 Data Type Each command must specify its transfer data type. Valid transfer data types are numeric, status, and open.

Numeric commands must specify the engineering units of the numeric data value (such as degrees F, inches, PSI, etc.). Unitless values (such as ratios) must be specified as unitless. Status word commands must specify the use of each bit or group of bits. Any reserved and user definable bits must be specified as such. Open commands must specify the contents and byte length of the open data. If the open data is used to represent complex data type (such as blanket data), the exact format of the data must be specified.

2.7.1.2.2.4 Support Requirement Each command of a command set must be specified as REQUIRED or OPTIONAL. REQUIRED commands MUST be supported by all implementations of the specific device. A control station can assume the support of required commands based on a specific DEVID. OPTIONAL commands maybe supported as desired by an implementation of a particular device. A control station cannot assume the support of optional commands.

2.7.1.2.2.5 Description Each command must be fully documented concerning its use. Information such as specific data validity requirements, multi-command access procedures, data zone usage, and any other special requirements must be fully explained by the command description.

2.7.1.2.3 SPI CCP Required Commands The SPI CCP requires that all approved command sets define an ECHO command and a Device/Command Set Revision command.

2.7.1.2.3.1 Echo Command The ECHO command is defined as an open data command consisting of 4 data bytes. The command is used to allow a control station to read and write any desired 4 byte data value. The value must be preserved by the device until power down or changed by the control station. The following command parameters are used to define the ECHO command:

Command Parameter	Description
Name	ECHO
Required	Yes
CMD1	$20
POLL CMD2	$20
SELECT CMD2	$21

Command Parameter	Description
Data Type	OPEN, 4 bytes in length, each byte can be any 8 bit data pattern (`$00-$FF`).
Description	Used to store any 4 byte value set by the control station. Value must be preserved by the tributary station until power-down or changed by the control station.

Table 17: Echo Command Parameters

2.7.1.2.3.2 Device Revision Command The Device/Command Set Revision command is defined as an open data command consisting of 4 data bytes. The data consists of ASCII characters which define the Device Revision Level. The command is a READ-ONLY command. The following command parameters are used to define the command:

Command Parameter	Description
Name	REVISION
Required	Yes
CMD1	`$20`
POLL CMD2	`$22`
SELECT CMD2	Not Supported
Data Type	OPEN, 4 ASCII bytes used as follows: +00: Major Rev Level 10's Digit +01: Major Rev Level 1's Digit +02: Minor Rev Level 10's Digit +03: Minor Rev Level 1's Digit
Description	Used to read a device's Device Revision Level.

Table 18: Device Revision Command Parameters

2.8 Layer 7, Application Layer

The OSI Model Application Layer is the topmost layer of the model. The main function of this layer is to allow the exchange of information by distributed application processes. It is at this layer that all underlying layer usage requirements are defined to form a complete protocol.

2.8.1 SPI CCP Machine Cell Serial Network Application

The SPI CCP has developed an OSI Model Application layer for use as a Machine Cell Serial Network. This application is for use by a primary plastics processing machine to communicate data with its auxiliary devices and was first defined fby the SPI CCP Version 3.x Auxiliary Device Protocol.

2.8.1.1 Application Scope The SPI CCP Machine Cell Serial Network was created to allow a primary plastic processing machine to communicate with its auxiliary devices. The purpose of the application is to allow the centralized setup and monitoring of the auxiliary devices in a single machine cell. The application does not provide for the direct control of the auxiliary devices by the primary plastics processing machine.

2.8.1.2 Network Topology and Underlying Layer Requirements The SPI CCP Machine Cell Serial Network is designed for a primary plastic processing machine and its auxiliary devices to communicate over a single serial based communication network. The following sections define the underlying OSI Model layer requirements.

2.8.1.2.1 SPI CCP Serial Physical Standard The SPI CCP Machine Cell Serial Network application requires the use of the SPI CCP Serial Physical Standard as the underlying physical layer protocol. This is the only physical layer protocol supported by this application. All devices connected to the network must support a default baud rate of 1200 bps. Optionally, baud rates of 2400, 4800, 9600, and 19.2k may also be supported. If a baud rate other than the default baud rate is to be used, all devices on the network must be physically set to the specified baud rate prior to any communication attempts.

2.8.1.2.2 SPI CCP Serial Data Link Standard The SPI CCP Machine Cell Serial Network application requires the use of the SPI CCP Serial Data Link Standard as the underlying data link layer protocol. This is the only data link layer protocol supported by this application. The primary plastics processing machine is set as the control station and all auxiliary devices are set as tributary stations. As the control station, the primary plastics processing machine is responsible for starting all communication with tributary stations and control of the communication link. The control station will initiate a communication session using POLL and SELECT requests based on specific SPI CCP Device Command Sets. The SPI CCP Application Standard does not define any procedures for communication session establishment or termination other than what is provided by the SPI CCP Data Link Standard. It is up to each specific control station implementation to determine which tributary devices it can address and support. When creating such an implementation, the following should be considered:

- Communication with specific tributary devices should be able to be enabled and disabled to allow a device to be disconnected without affecting the communication network.
- The control station should be able to report device communication errors to its application process (for example, informing the control station operator).
- The control station can assume the support of required commands based on a device's DEVID.

If commands marked as optional are attempted, the control station should be prepared to handle errors if the commands are not supported by the auxiliary device. As tributary stations, auxiliary devices are expected to support all required commands based on their defined device type DEVID.

2.8.1.2.3 Network Layer The SPI CCP Machine Cell Serial Network requires no specific network layer protocol. Most control and tributary station implementations will not require a network layer. If such a layer is required, it is up to the specific application to define its specifications.

2.8.1.2.4 Transport Layer The SPI CCP Machine Cell Serial Network requires no specific transport layer protocol. It is up to the control and tributary station application to present valid data in the proper transmission order to the underlying communication layers. Data must always be presented as a static set of data so that the first byte of data belongs with the last byte of data.

2.8.1.2.5 Session Layer The SPI CCP Machine Cell Serial Network requires no specific session layer protocol. The control station application must define its method of communication establishment and error processing. Most data exchange, data synchronization, and data semaphore processing is defined at the presentation level in specific device command sets.

2.8.1.2.6 SPI CCP Presentation Standard The SPI CCP Machine Cell Serial Network requires the use of the SPI CCP Presentation Standard. This standard specifies device specific command sets which define the syntax and format of data to be communicated. The primary plastics processing machine will use these command sets to communicate with the auxiliary devices in its cell. It is up to the control station

application to define which devices it supports, how data is exchanged (when data is POLLed and SELECTed), and how errors are processed.

2.8.2 SPI CCP Host Computer Serial Network Application

The SPI CCP has developed an OSI Model Application layer for use as a Host Computer Serial Network. This application is for use by a host computer to communicate data with factory floor plastics processing equipment.

2.8.2.1 Application Scope The SPI CCP Host Computer Serial Network was created to allow a host computer system to communicate with plastics processing equipment. The protocol is designed to allow the host computer to monitor process data, collect alarm information, read and write setup parameter files, and exchange various other information. Unlike the SPI CCP Machine Cell application, this application is concerned with communicating with the primary machines of a group of cells. Cell data is obtained for the host computer from the primary cell machines. Typically, an SPI CCP Host Computer Serial Network application will consist of the host computer and a group of primary cell machines such as injection molding machines. An application may also include auxiliary machines such as plant wide loading or handling systems.

2.8.2.2 Network Topology and Underlying Layer Requirements The SPI CCP Host Computer Serial Network is designed for a host computer and primary plastic processing machines to communicate over a single serial based communication network. The following sections define the underlying OSI Model layer requirements.

2.8.2.2.1 SPI CCP Serial Physical Standard The SPI CCP Host Computer Serial Network application requires the use of the SPI CCP Serial Physical Standard as the underlying physical layer protocol. This is the only physical layer protocol supported by this application. All devices connected to the network must support a default baud rate of 9600 bps. Optionally, other baud rates may also be supported. If a baud rate other than the default baud rate is to be used, all devices on the network must be physically set to the specified baud rate prior to any communication attempts.

2.8.2.2.2 SPI CCP Serial Data Link Standard The SPI CCP Host Computer Serial Network application requires the use of the SPI CCP Serial Data Link Standard as the underlying data link layer protocol. This is the only data link layer protocol supported by this application. The host computer is set as the control station and all primary cell machines or auxiliary devices are set as tributary stations. As the control station, the host computer is responsible for starting all communication with tributary stations and control of the communication link. The control station will initiate a communication session using POLL and SELECT requests based on specific SPI CCP Device Command Sets. The SPI CCP Application Standard does not define any procedures for communication session establishment or termination other than what is provided by the SPI CCP Data Link Standard. It is up to each specific control station implementation to determine which tributary devices it can address and support. When creating such an implementation, the following should be considered:

- Communication with specific tributary devices should be able to be enabled and disabled to allow a device to be disconnected without affecting the communication network.
- The control station should be able to report device communication errors to its application process (for example, informing the control station operator).
- The control station can assume the support of required commands based on a device's DEVID.

If commands marked as optional are attempted, the control station should be prepared to handle errors if the commands are not supported by the tributary station. As tributary stations, primary and auxiliary devices are expected to support all required commands based on their defined device type DEVID.

2.8.2.2.3 Network Layer The SPI CCP Host Computer Serial Network requires no specific network layer protocol. Some host computer applications may require a network layer protocol to transfer information across other application specific networks. In such cases, it is up to the host computer application to supply this layer.

2.8.2.2.4 Transport Layer The SPI CCP Host Computer Serial Network requires no . specific transport layer protocol. It is up to the control and tributary station application to present valid data in the proper transmission order to the underlying communication layers. Data must always be presented as a static set of data so that the first byte of data belongs with the last byte of data.

2.8.2.2.5 Session Layer The SPI CCP Host Computer Serial Network requires no specific session layer protocol. The control station application must define its method of communication establishment and error processing. Most data exchange, data synchronization, and data semaphore processing is defined at the presentation level in specific device command sets.

2.8.2.2.6 SPI CCP Presentation Standard The SPI CCP Host Computer Serial Network requires the use of the SPI CCP Presentation Standard. This standard specifies device specific command sets which define the syntax and format of data to be communicated. The host computer will use these command sets to communicate with the primary and auxiliary cell devices connected to the network. It is up to the control station application to define which devices it supports, how data is exchanged (when data is POLLed and SELECTed), and how errors are processed.

3 Appendices

3.1 Abbreviations and Acronyms

The following are various abbreviations and acronyms used by this document.

- **ADD ADDress** — An 8-bit quantity used during the connection establishment procedures to identify a particular tributary station. Used in conjunction with the DEVID. The actual address values are selectable at the tributary stations.
- **CMD1 CoMmanD1** — An 8-bit quantity used during the connection establishment procedures to specify the most significant command byte. Used in conjunction with CMD2 to specify a particular action to be taken or data to be communicated.
- **CMD2 CoMmanD2** — An 8-bit quantity used during the connection establishment procedures to specify the least significant command byte. Used in conjunction with CMD1 to specify a particular action to be taken or data to be communicated. An even CMD2 (least significant bit is reset) indicates a POLL request. An odd CMD2 (least significant bit is set) indicates a SELECT request.
- **DEVID DEVice IDentifier** — An 8-bit quantity used during the connection establishment procedures to identify a particular tributary station device type. Used in conjunction with the ADD. The actual values of the DEVID are assigned by the SPI/CCP.
- **EOB End Of Block** — An acronym used to delimit the end of a message block.
- **ERR ERRor byte** — An 8-bit quantity used as a prefix for negative acknowledgments to indicate why the acknowledgment was negative.
- **ms** — Milliseconds, $\frac{1}{1000}$ of a second.
- **Network Bus** — The physical bus media which extends from the control station. All tributary stations are connected to the bus via Unit Stubs.

- **Timer "E"** — Used by a device to wait after the last character has been transmitted before the device begins its own data transmission. The timer is started when a receiver detects the last character has been transmitted by a sending device. The timer is never restarted or stopped. Timeout of Timer "E" indicates that the waiting device is now able to begin transmission.
- **Unit Load** — A hypothetical quantity representing the DC loading that passive generators and receivers place on the communication circuit. You must examine the hardware specifications for your communication chip(s) set to define the unit loading produced by your transceiver.
- **Unit Stub** — The physical media used to connect a tributary station to the network bus.
- **<nn>** — nn is a hexadecimal constant.

3.2 Glossary

The following are various terms used by this document.

- **Aborts** — When the sending station desires to end a block or transmission in an unusual manner such that the receiving station disregards that portion of the block or transmission that has been received.
- **Alternating Acknowledgments** — Used as a reply to a master station that a valid message has been received and accepted. There are two valid alternating acknowledgment character sequences, DLE '0' (ACK0) and DLE '1' (ACK1). ACK1 is used as the acknowledgment for the first transmission block received, and the acknowledgment is alternated between ACK0 and ACK1 on subsequent blocks. ACK0 is also used as the acknowledgment by a tributary to the control station's SELECT request supervisory sequence.
- **ANSI (American National Standards Institute)** — The main standards development group in the USA. ANSI is a non-profit non-government group composed of over 1000 trade organizations, professional societies, and companies. It is the USA's member body to ISO.
- **ASCII** — American National Standard Code for Information Interchange. Specifies the set of 128 control and data characters defined as a standard data communications code in American National Standard Code for Information Exchange. Each character is specified in seven binary bits, with the first 32 characters representing control and special action characters. See the ASCII chart following this glossary.
- **Baud** — A unit of signaling speed, the number of signal events per second. If each signal event represents only one bit condition, the baud is equal to bits per second.
- **Bit (Binary Digit)** — The smallest unit of information in a binary system. A bit represents a choice between 1 and 0.
- **Centralized Operation** — A control discipline for multipoint data communications links by which transmission may occur between the control station and any one of the tributary stations, but not between tributary stations themselves.
- **Communications Facility** — The actual medium used to provide communication circuits. Some examples may be wire, cable, radio, fiber optics.
- **Contention** — Occurs when two or more stations attempt a transmission at the same time.
- **Control Station** — The station on a data communications link with the overall responsibility for monitoring the communications link and assuring the links orderly operation. The control station has the responsibility for initiating recovery procedures in the event of abnormal conditions on the link.
- **CRC-16 (Cyclic Redundancy Check 16)** — An error detection scheme designating the bits added to the end of a message or transmission block to facilitate error control. The '16' specifies use of a degree-16 polynomial generator.

- **Data Communications Circuit** — A means of two-way communication between two points.
- **Data Communication Control Character** — A functional ASCII character intended to control or facilitate transmission of information over communication networks. There are ten control characters defined by ASCII which will be used in this protocol: EOT, SOH, STX, ETX, ACK, NAK, ENQ, ETB, SYN, DLE.
- **Data Communication Control Procedure** — The means used to control the orderly communication of information between stations on a data communication link.
- **Data Communication Link** — The logical association of two or more stations interconnected by the same data communication circuit, including the communication control capability of the interconnected stations.
- **Data Communication Station** — The independently controllable configuration of logical elements, from or to which messages are transmitted on a data communication link. It includes those elements (data communication equipment, intermediate terminal equipment) that control the message flow on the link via data communication control procedures.
- **Data Integrity** — A performance measure based on the rate of undetected errors.
- **Delimiters** — Control characters used to define the extent of a particular sequence of characters (for example delimiting the entry and exit of transparent data mode).
- **EIA (Electronic Industries Association)** — A standards organization specializing in electrical interfacing.
- **Half Duplex** — Transmission in either direction, but not both directions simultaneously.
- **Interrupt** — When a receiving station wishes to cause the sending station to stop sending, either temporarily or permanently.
- **ISO** — International Standards Organization.
- **Mark** — presence of a signal. A mark impulse is equivalent to a set (1) bit.
- **Master Station (Control Station)** — A station that has control of the data communication link at a given instant. The assignment of master status to a given station is temporary and is controlled by the procedures set forth in the categories described in this standard. Master status is normally conferred upon a station so that it may transmit a message, but a Station need not have a message to send to be nominated master.
- **Message** — A sequence of characters arranged for the purpose of conveying information from an originator to one destination (or address). It contains the supplementary information in a heading.
- **Message Heading**— The part of a message containing all components preceding the text. A message heading is preceded by the TSOH sequence and followed by the TSTX sequence (delimited by TSOH and TSTX).
- **Message Text** — The part of a message beginning with the first character following the TSTX sequence and followed by the TETX sequence.
- **Multipoint Link** — A data communication link connecting two or more stations.
- **Polling** — A technique for inviting a station to transmit messages at a given time. One station is designated as a control station to invite tributary station(s) to transmit.
- **Polling Supervisory Sequence** — A supervisory sequence that performs the polling function.
- **Prefix** — A sequence of characters (other than communications control characters) used in a supervisory sequence to define or qualify the meaning of the supervisory sequence.

- **Protocol** — A formal set of conventions governing the format and relative timing of data exchange between two or more communicating systems.
- **Recovery Procedure** — Control procedures used to restore normal operation to a data communication link after unusual or abnormal events have occurred.
- **Reply** — A supervisory sequence by which a slave station informs the master station of its operational condition or status, or the status of a received message or message block.
- **Reply-Request Supervisory Sequence** — A supervisory sequence used by a master station to request a reply from a slave station.
- **Reserved Character(RES)** — A character reserved for later definition by SPI. At this time the character must be <20>.
- **Responder** — The station that is responsible for supplying reply messages to the initiator.
- **Selection** — A technique for assignment of slave status to a station on a data communication link.
- **Selection Supervisory Sequence** — A supervisory sequence that performs a selection function.
- **Slave Station** — A station that has been selected to receive a transmission from the master station. The assignment of slave status is temporary, under control of the master station, and continuous for the duration of a transmission.
- **Space** — Absence of a signal, used to indicate a reset (0) binary condition.
- **SPI (Society of Plastics Industry)** — A professional society consisting of representatives of companies involved in all aspects of the plastics industry.
- **Station Identification** — A sequence of characters used to identify a particular station.
- **Start Bit** — The first bit in each character, normally a space, which causes the receiving equipment to prepare for the reception of a character.
- **Stop Bit** — The last bit or element in each character, normally a mark, to which is assigned a minimum duration, during which the receiving equipment is returned to its rest condition in preparation for the reception of the next character.
- **Supervisory Sequence** — A sequence of communication control characters and possibly other characters that performs a defined control function. A supervisory sequence may contain a prefix together with any required delimiters.
- **Timer "A"** — ANSI x3.28 response timer. Used by the sending station to protect against an invalid response or no response. Timer A is started after transmitting any ending character where a response is expected (such as ENQ, TETB, or TETX). Timer A is stopped by receipt of an entire valid reply (such as ACK0, NAK,etc.). Timer A is never restarted. A recovery procedure is to be executed if Timer A times-out.
- **Timer "B"** — ANSI x3.28 receive timer. Used by a receiving station to protect against nonrecognition of TETB or TETX sequences. Timer B is started upon receipt of the start of block or text (TSOH or TSTX). Timer B may be restarted upon receipt of characters. Timer B is stopped by receipt of a valid terminating character or character sequence (such as TETX, TETB, ENQ, etc.). A recovery procedure is to be executed if Timer B times-out.
- **Timer "D"** — ANSI x3.28 non-activity timer. Used by all stations on the communication link to determine no activity on the link. Timer D is started or restarted upon receipt of any character. Timer D is stopped upon receipt or transmission of EOT. A recovery procedure is to be executed if Timer D times-out.

- **Time-Out** — Occurs when one of the ANSI timers (A,B, or D) counts down to zero before being stopped or reset.
- **Transmission** — The entirety of the data transmitted between the master station and slave station for the period of their uninterrupted assignment of such status.
- **Transmission Block** — A group of characters transmitted as a unit. A transmission block may contain a message, a portion of a message, or combinations thereof.
- **Transparent Data** — Allows the transmission of all 8-bit codes as data in the text part of a message.
- **Tributary Station** — A station on a data communication link that is not the control station.
- **Two-way Alternate Transmission** — A type of transmission through which messages or transmission blocks may be sent in either one direction or the other, but not both directions simultaneously.

3.3 ASCII Character Set

The following table contains the 7 bit ASCII character set

LSD\MSD	0	1	2	3	4	5	6	7
0	NUL	DLE	SPC	0	@	P	‘	p
1	SOH	DC1	!	1	A	Q	a	q
2	STX	DC2	"	2	B	R	b	r
3	ETX	DC3	#	3	C	S	c	s
4	EOT	DC4	$	4	D	T	d	t
5	ENQ	NAK	%	5	E	U	e	u
6	ACK	SYN	&	6	F	V	f	v
7	BEL	ETB	’	7	G	W	g	w
8	BS	CAN	(	8	H	X	h	x
9	HT	EM	)	9	I	Y	i	y
A	LF	SUB	*	:	J	Z	j	z
B	VT	ESC	+	;	K	[	k	{
C	FF	FS	,	<	L	\	l	|
D	CR	GS	--	=	M	]	m	}
E	SO	RS	.	>	N	^	n	~
F	SI	US	/	?	O	_	o	DEL

Table 19: 7-bit ASCII Character Set

The following are descriptions of the ANSI control characters and control character sequences.

- **EOT End Of Transmission <04>** — The EOT character performs the following functions:
  1. Cancels any previous master/slave relationship.
  2. Sent by the master station after completion of a message transfer phase to normally end a transmission.
  3. Sent by the master station prior to completing the message transfer phase to Cause a sending station abort.
  4. Sent by the slave station in place of a normal response to cause a termination interrupt.

  The EOT character must never have a prefix.

- **ENQ ENQuiry <05>** — The ENQ character is used to illicit a response from station. The ENQ calls for a reply indicating the slave stations readiness to receive. The ENQ character is also used as the last character in a polling or selection sequence. Receipt of the ENQ character by a slave station calls for the slave station to retransmit its last reply (ACK0, NAK, etc.).
- **ACKN Acknowledge N** — ACKN is represented by the <10><30> for ACK0 and <10><31> for ACK1 control sequences. The ACKN is used by a receiver as an affirmative reply. ACK1 is used as an affirmative reply to the first block of a message, and affirmative replies to subsequent blocks alternate between ACK0 and ACK1. ACK0 is also used as the reply from tributary to the control station in response to a SELECT supervisory request.
- **DLE Data Link Escape <10H>** — The DLE character is used to change the meaning of the character. It is used exclusively for providing additional control functions. It is used in the following control functions:

  1. Alternating Acknowledgments.
  2. Transparent block delimiting sequences (TSOH, TSTX, TETX, and TETB).
  3. Transparent DLE (a DLE, <10>, character in a transparent text sequence is represented by DLE DLE).
  4. The block abort function.
  5. The reverse interrupt function.
  6. The synchronous idle function.

- **NAK Negative-AcKnowledgment <15>** — The NAK character is transmitted by a slave as a negative response to a sending station. The NAK may contain a prefix indicating why the message or transmission block was rejected.

  The NAK performs the following functions:

  1. NAK indicates that the selected station is not ready to receive.
  2. Is transmitted by a slave to indicate that the last message or transmission block was not accepted and the slave is again ready to receive.

  A prefix will be included indicating why the transmission was rejected.

- **TSOH Transparent Start of Header (DLE SOH)** — Used at the beginning of a sequence of characters that constitutes a machine address or routing information. The TSTX sequence terminates the header.
- **TSTX Transparent Start of Text (DLE STX)** — This control sequence is used to indicate the start of transparent text. It also serves to terminate a transparent header sequence if one precedes it. The TSTX places the data link in a condition to process transparent data. If a message consists of multiple transmission blocks, the TSTX sequence is used to start all subsequent blocks.
- **TETB Transparent End of Block (DLE ETB)** — The TETB sequence is used to terminate a block of transparent data that is not the last block of the message. It signals that the next two following characters are the CRC characters and that the slave should transmit a reply. TETB returns the data link to a non-transparent condition.
- **TETX - Transparent End of Text (DLE ETX)** — The TETX control sequence indicates the end of a transparent message. It signals the slave device that the next two following characters are the CRC characters and that the slave is expected to transmit a reply. This sequence is used to terminate the last block of a message and causes the data link to return to a non-transparent condition.
- **Block Abort (DLE ENQ)** — The sending station, while sending data, but before the end of the block, decides to end the block. The receiving station will disregard all data received for that block. Both stations return to non-transparent mode. The NAK response is the only valid response to the aborted block.

- **Sending Station Abort (EOT)** — The sending station, after completing a transmission block (DLE ETB or DLE ENQ), and after receipt of a valid response (ACK0, NAK,etc.) or timer-A time-out, prematurely terminates the transmission to the receiving station by transmitting an EOT. The transmission of the EOT resets all stations on the communication link, and returns all stations to the non-transparent control mode.
- **Termination Interrupt (EOT)** — The receiving station, deciding that it is unable to continue to receive data, transmits an EOT instead of one of the normal responses (ACK0, NAK, etc.). This transmission indicates a non-acknowledgment of the data block, resets all stations on the communication link, and returns all stations to the non-transparent control mode.
- **Reverse Interrupt (DLE <3C>)** — The control station, while acting as a receiving station, may decide to interrupt the current communication session to poll or select another station. The station may only generate a reverse interrupt after receipt of a valid CRC. The reverse interrupt is assumed by the sending station to be a valid acknowledgment to the transmitted block. The sending station should then transmit an EOT to abort the communication session as soon as it can without losing any data it may have stored in buffers. It then must be able to receive a transmission.
- **Synchronous Idle Sequence (SYN) or (DLE SYN)** — Used as a "time-fill" sequence when a sending station is unable to send its normal data. The SYN sequence may be added at any time during the transmission except:

  1. In a DLE sequence.
  2. Between DLE ETB or DLE ETX and the following CRC characters.

3.4 CRC Calculation Program Example

The following is a C language program example for calculating a CRC-16 value. The program uses the following polynomial in this calculation: $CRC_{16} = X^{16} + X^{15} + X^2 + 1$ The `InitCRCTable()` function is to be called once to build the CRC lookup table. The `Addcrc()` function is called to add characters to a CRC calculation register. The register should be reset to 0 before the first character of a sequence to be checked is added.

```
/* This C language file contains routines used to calculate a CRC-16.
   The polynomial used is (X^16 + X^15 + X^2 + 1).
   The first function will generate a lookup table required when calculating a CRC.
   The second function is called to add a BYTE to a crc value.
   The crc value should be reset to 0 and then data bytes may be added.
*/

/* *** Define the CRC-16 Polynomial (reflected) *** */
/* Note: x^16 + x^15 + x^2 + 1 in normal form is 0x8005; the reflected
   (LSB-first, right-shift) form is 0xA001, which matches the code's update method. */

#define CRC_POLY 0xA001u

/* *** Define CRC lookup table array *** */
static unsigned short crc_table[256];

/*
   InitCRCTable();

   This function is to be called ONCE at initialization to create the CRC lookup table.
   Historically this was declared FAR for segmented memory; in modern C it is a normal function.
*/
void InitCRCTable(void)
{
    unsigned short i, j;
```

```
    unsigned short crc;

    for (i = 0; i < 256; i++) {
        crc = i; /* <-- use the table index as the starting byte */
        for (j = 0; j < 8; j++) {
            if (crc & 0x0001u) {
                crc = (crc >> 1) ^ CRC_POLY;
            } else {
                crc >>= 1;
            }
        }
        crc_table[i] = crc;
    }
}

/*
   AddCRC (crc, value)
       unsigned short *crc - a pointer to the 2-byte CRC register
       unsigned char value - the value to add to the CRC

   This function is called to add a byte to the CRC.
*/
void AddCRC(unsigned short *crc, unsigned char value)
{
    unsigned short temp;

    /* Mix in the next input byte (reflected/LSB-first method). */
    temp = (unsigned short)(*crc ^ (unsigned short)value) & 0x00FFu;

    /* Shift and table-lookup */
    *crc = (unsigned short)((*crc >> 8) ^ crc_table[temp]);
}
```

3.5 References

The following documents, articles, and volumes were used in the development of this document and may be referenced for further information.

ANSI X3.28-1976, American National Standard Procedures for the Use of the Communication Control Characters of American National Standard Code for Information Interchange in Specified Data Communication Links, American National Standards Institute, 1975.

Richards, T., An Overview of the Society of the Plastics Industry Committee on Communication Protocol, SPI, 1989.

Richards, T., The SPI Communication Protocol: A Practical Approach to Computer Communication in the Molding Shop, SPI, 1994.

Society of the Plastics Industry Committee on Communication Protocol, Communication Protocol Manual V3.04, SPI, 1987 - 1993.

Society of the Plastics Industry Committee on Communication Protocol, Meeting Minutes, SPI, 1987 - 1994.

Stallings, W., Handbook of Computer Communications Standards, Indianapolis: Howard W. Sams & Co., 1987